Predicción de consumo de agua mediante redes neuronales de funciones base radiales evolutivas

V. M. Rivas, M.G. Arenas
Departamento
de Informática
Universidad de Jaén
EPS de Jaén
23071 Jaen
vrivas@ujaen.es, mgarenas@ujaen.es

J. Merelo, A. Prieto
Dept. de Arquitectura y
Tecnología de Computadores
Universidad de Granada
ETS Ingeniería Informática
18071 Granada
jmerelo@geneura.ugr.es, aprieto@ugr.es

Resumen

Este trabajo describe el algoritmo EvRBF así como los resultados que obtiene en la predicción de la serie temporal propuesta a concurso dentro del SICO'2007[1]. EvRBF es un algoritmo evolutivo que permite diseñar Redes Neuronales de Funciones Base Radiales. Para ello, utiliza operadores evolutivos diseñados específicamente para el trabajo con este tipo de redes. Entre sus características destaca que evita partir de una determinada topología de red, no impone restricciones con respecto al número de neuronas que deben tener las redes y utiliza exclusivamente información sobre la capacidad de generalización de cada red, así como su tamaño, para evaluar cada individuo de la población. Bajo estas condiciones, EvRBF es capaz de encontrar redes no sobredimensionadas en cuanto a su tamaño y con una alta capacidad de generalización.

1. Introducción

Las redes neuronales de funciones base radiales (RNFBR, Broomhead y Lowe [18]) se caracterizan porque las funciones de activación de sus neuronas ocultas son funciones base radiales, normalmente funciones gausianas. Estas redes se componen de dos capas de neuronas; cada neurona se conecta con todas las neuronas de la siguiente capa y la información fluye en un solo sentido, desde la entrada a la salida.

La salida de una RNFBR es la mostrada en la ecuación 1.

$$s_j(\vec{x_k}) = \lambda_{0j} + \sum_{i=1}^{p'} \lambda_{ij}\phi_i(\vec{x}, \vec{c_i}, \vec{r_i}) \qquad (1)$$

donde

$$k = 1..p, j = 1..n', s_j \in \mathbb{R}, \vec{x_k} \in \mathbb{R}^n \qquad (2)$$

y

- ϕ_i es la FBR asignada a la neurona oculta i;
- λ_{0j} es el sesgo;
- λ_{ij} representa el peso entre la neurona oculta i y la neurona de salida j;
- $\vec{c_i}$ y $\vec{r_i}$ se denominan, respectivamente, el *centro* y los *radios* de las FBR;
- n y n' son las dimensiones de los espacios de entrada y salida, respectivamente;
- p' es el número de neuronas ocultas, y p el número de patrones a los que s_j se va a aplicar.

[1] SIMPOSIO DE INTELIGENCIA COMPUTACIONAL, realizado dentro del II Congreso Español de Informática

La principal ventaja de las RNFBR radica en que los pesos y sesgos óptimos (es decir, λ_{0j} y λ_{ij}) se pueden computar eficientemente para un conjunto de salidas deseadas si previamente se han fijado las neuronas ocultas (número, centros y radios). Las ecuaciones 3 y 4 muestran cómo se realiza dicho cálculo.

$$F = A\lambda \tag{3}$$

cuya solución se expresa en la ec. 4:

$$\lambda = A^{-1^*}F \tag{4}$$

donde F es el conjunto de salidas deseadas; A es la *matriz de diseño* (A_{ij} representa la salida de la neurona oculta j cuando el patrón de entrada i se aplica a la red); λ es el conjunto de pesos y sesgos (siendo λ_{jk} el peso de la conexión entre la neurona oculta j y la neurona de salida k); y A^{-1^*} representa la *pseudo-inversa* de la matriz A.

Los sesgos y pesos calculados mediante A^{-1^*} consiguen minimizar el error cuadrático medio con respecto al conjunto de entradas usado para calcularlos. Si se utilizan menos neuronas ocultas que número de patrones a ser aproximados (es decir., $p' << p$), se puede utilizar descomposición de valores singulares (SVD)[23] o cualquier método de descenso de gradientes para calcular A^{-1^*}.

Este trabajo muestra los resultados más recientes obtenidos con EvRBF, método diseñado para construir de forma totalmente automática redes neurolanes FBR que resuelve problemas de clasificación. A diferencia de la mayoría de los métodos ([15]), EvRBF es un algoritmo evolutivo que establece todos los parámetros que conforman la red. Por ello, include operadores genéticos específicos para operar sobre la estructura de RNFBR: número de neuronas ocultas, así como centros y radios de cada una de ellas. Además, de partida no impone ningún límite al número de neuronas que pueden tener las redes por lo que el crecimiento de las mismas es guiado por su capacidad para resolver cada problema concreto de clasificación.

EvRBF no usa alfabetos binarios para la codificación del cromosoma, por ello la *Teoría de esquemas* [12] no puede aplicársele. Sin embargo, representaciones más complejas de los cromosomas encuentran su soporte científico en la *Teoría de análisis de Formas* [24] que amplía la anterior mediante la equivalencia de clases, en vez de esquemas. Así, la Teoría de Análisis de Formas asegura que la utilización de operadores que no dependen de la representación explícita del cromosoma, permite la confección de algoritmos evolutivos independientes. Estos algoritmos tienen un comportamiento similar al de los algoritmos genéticos estandar.

El resto del artículo se ha organizado como sigue: la sección 2 muestra una breve revisión de métodos diseñados para la construcción de RNFBR. La sección 3 describe el algoritmo EvRBF, prestando una especial atención a los operadores creados para el tratamiento de estas redes. La sección 4 muestra el conjunto de experimentos llevados a cabo para determinar la predicción realizada por EvRBF en la serie temporal propuesta a concurso. Finalmente, la sección 5 describe las conclusiones y futuras líneas de trabajo.

2. Estado del arte

Los métodos existentes para el diseño automático de RNFBR pueden dividirse en evolutivos y no evolutivos.

2.1. Métodos no evolutivos.

Según [29], los métodos no evolutivos pueden ser clasificados en función del número de pasos necesarios para entrenar la red.

Los métodos de un paso seleccionan puntos del conjunto de patrones disponibles para colocarlos como centros de la red; los radios se fijan a un valor preestablecido; a continuación, se calculan los pesos. Las *Máquinas de Soporte Vectorial* (SVM) [3] se convierten en RNFBR cuando se usan funciones gaussianas como funciones de activación y son entrenadas mediante este método.

Los métodos de dos pasos son los más usados. El primer paso utiliza cualquier clase de algoritmo capaz de seleccionar puntos del espacio de entradas; dichos puntos serán usados como

los centros de las neuronas y pueden diferir de los presentes en el conjunto de entrenamiento. Métodos como agrupamiento, [20], LVQ [28], y árboles de decisión [16] han sido usados para establecer los centros.

El establecimiento de los radios (tarea crítica según [1]) también se realiza en este paso. Habitualmente, los radio se establecen a un valor proporcional a la media de las distancias desde cada neurona a las k más cercanas. El ratio de ponderación varía entre 0,5 y 1,75, según el autor; e igualmente, k varía desde 1 a $p' - 1$ (siendo p' el número de centros a establecer).

Una vez fijados centros y radios, el sgundo paso calcula el conjunto de pesos correspondiente.

Los métodos de tres pasos [29] se diseñan para modificar de forma iterativa los centros y los radios, usando para ello un algoritmo de retropropagación del error cometido por la red. Así, el primer paso establece los centros y radios; el segundo calcula los psos; y el tercero utiliza algún algoritmo de descenso de gradientes para afinar el valor de centros y radios.

En general, los métodos anteriores tienen dos inconvenientes fundamentales: el tamaño de la capa oculta debe ser establecido a priori, y el método para afinar pesos, centros y radios puede quedar atrapado en un mínimo local.

Los métodos híbridos pueden modificar el número de neuronas que componen la capa oculta. Así, *los métodos incrementales* comienzan con un número reducido de neuronas (o incluso una sola) y añaden nuevas de forma que se minimice el error cometido por la red. Métodos que pertenecen a este grupo son: OLS (*Orthogonal Least Squares, [8]*), *OLS regularizado* [9], RAN (*Resource Allocating Networks, [21]*), y CSG (*Cell Structure Growing, [11]*). Los mecanismos que determinan cuándo y cómo debe crecer hacen diferentes a cada un o de los métodos.

Los métodos decrementales o de poda actúan a la inversa. Comienzan con una red sobreespecificada y eliminan neuronas de forma iterativa. Así, en [17] se utiliza la medida de mínima longitud descriptiva (*Minimum Description Length, MDL*) para seleccionar el menor conjunto de neuronas que mejor describen el conjunto de datos de entrenamiento.

Los métodos GAP-RBF y FGAP-RBF (*Growing and Pruning RBF* y *Fast Growing and Pruning RBF*, respectivamente) presentados en [32] pertenecen también a este tipo de métofos. Ambos se basan en el denominado *filtro de Kalman extendido* (EKF) para determinar si se debería añadir o quitar una neurona para, posteriormente, recalcular el valor de los pesos.

Tanto los métodos incrementales como los decrementales tienden a ser muy restrictivos, dando más importancia al número final de neuronas que a la capacidad de generalizacion de la red, obteniendo normalmente redes subóptimas.

2.2. Algoritmos evolutivos

Los algoritmos evolutivos han sido aplicados al diseño de redes neuronales de muchas formas [31]. Mientras la mayoría de los métodos se centran en optimizar sólo una parte de la red (pesos o estructura), otros intentan omptimizar todos los parámetros de la misma, como en [19] para redes LVQ, o [7] para perceptrones multicapa.

Harpham et al. ([15]) han revisado algunos de los métodos más conocidos de aplicación de algoritmos evolutivos al diseño de redes FBR. Como característica general, concluyen que los métodos se concentran, normalmente, en una sóla característica de modo que se recurre a métodos no evolutivos para seleccionar la mayor parte de los parámetros que conforman la red, dejando al algoritmo evolutivo la tarea de establecer sólo los centros o los radios.

Cronológicamente, primeros trabajos [5, 30] usaban codificación binaria y estaban restringios por el número de neuronas ocultas, que tenían que ser fijadas a priori. Ambos métodos se diferencian en la manera en que construyen la red. Así, Carse [5] trabajaba con una población de redes, mientras que Whitehead [30] lo hacía con una población de las neuronas (de esta forma, sólo una RNFBR era construida y evaluada) que competían pero también cooperaban para encontrar la red óptima.

Un trabajo posterior, [4], presentaba un algoritmo, basado en representación numérica,

con algunas ventajas, como operadores rápidos, optimización dinámica del tamaño de la capa oculta, fácil cómputo de la función de evaluación, y rápida convergencia a una solución válida. Sin embargo, la imposición empírica de un límite al número de neuronas, la optimización sólo de centros y no de radios para las FBR, y la mala definición de un término de penalización para el número de veces que un individuo puede reproducirse, representan las principales desventajas a tener en cuenta para este algoritmo.

Recientes aplicaciones de algoritmos evolutivos al diseño de RBFNN [13, 25] intentan superar las desventajas de los métodos precedentes. González [13] comienza con un método de agrupamiento mejorado para la inicialización de las redes, y después utiliza operadores de mutación, combinados con un método local de la búsqueda, para buscar eficientemente los parámetros de la RNFBR. Rivera [25] ofrece unpunto de partida diferente, así, varias neuronas compiten y son modificadas mediante *evolución digusa*, esto es, usando una tabla de reglas difusas que especifican que operador debe ser aplicado a cada neurona para mejorar su comportamiento.

La mayoría de los métodos recientes todavía se centran en un sólo aspecto al configurar una red. Así, Ros y otros. ([26]) presentaba un método para inicializar automáticamente RNFBR. El método se basa en *Quick Supervised Clustering and Supervised Fuzzy C-Means* (QSC y FCMS, respectivamente), trabajando en colaboración. Aunque el método puede encontrar un buen sistema de neuronas iniciales, requiere que se le imponga un número máximo de neuronas y termina delegando en cualquier otro método para mejorar la red y para fijar sus anchuras.

El algoritmo que se presenta en este trabajo trata de estimar en su totalidad la RNFBR que soluciona un determinado problema. Es un algoritmo rápido (a pesar de ser un algoritmo evolutivo), que incluye operadores específicos y utiliza una función de evaluación muy simple dirigida por la capacidad de generalización de las redes en su tarea de búsqueda.

3. El algoritmo EvRBF

EvRBF es un procedimiento iterativo en el cual la creación y evaluación de nuevas generaciones de individuos lleva a encontrar RNFBR con buena capacidad de la generalización. Es un algoritmo de estado estacionario, que incluye elitismo. El tamaño de los individuos es variable, si bien el tamaño de la poblacion es constante durante todo el proceso.

3.1. Operadores genéticos

EvRBF incluye operadores diseñados específicamente para manejar RNFBR. Algunos operadores están diseñados para escapar de mínimos locales, mientras que otros intentan intentan afinar soluciones previamente encontradas.

Los operadores de EvRBF pueden ser divididos en tres grupos: recombinación o cruce, modificación del centro y de los radios, y modificación del tamaño de la capa oculta. La recombinación es útil como mecanismo para compartir información entre individuos y es crucial en modelos cooperativos, como RBFNN o los mapas autoorganizativos. La modificación de centros, radios y número de neuronas pertenecen a los denominados *operadores de mutación*. Intentan introducir cierta variabilidad en el conjunto de genes, de modo que se exploren nuevas áreas de búsqueda. Los siguientes apartados muestran estos operadores agrupados en dichas categorías.

3.1.1. Recombinación: X_FIX y X_MULTI

Ambos operadores intercambian información entre los individuos para localizar los *bloques constructivos* de la solución.

X_FIX reemplaza una secuencia de neuronas de la red R_1 por otra del mismo tamaño tomada de R_2, como muestra la fig. 1

Por su parte, el operador X_MULTI reemplaza con probabilidad p_{x_multi} cada neurona oculta de la red R_1 por una neurona seleccionada aleatoriamente y proveniente de la red R_2 (ver fig. 2). Este operador se correspndería con el entrecruzamiento uniforme en un algoritmo genético de cromosomas binarios.

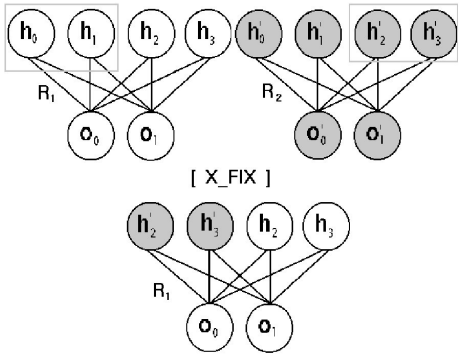

Figura 1: Aplicación e X_FIX: se intercambian neuronas sin que quede afectado el tamaño de la red resultante, R_1.

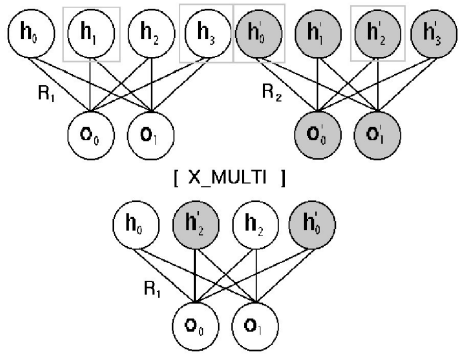

Figura 2: Aplicación de X_MULTI: neuronas de R_1 se reemplazan por neuronas de R_2, sin que cambie el tamaño de la red final, R_1.

3.1.2. Modificación de centros y radios: C_RANDOM y R_RANDOM

Estos operadores utilizan la aleatoriedad para aumentar la diversidad entre los individuos de la población; así, generan nuevos individuos de modo que los mínimos locales pueden ser evitados. El operador de C_RANDOM modifica los centros de las FBR. El número exacto de neuronas afectadas por este operador viene determinado por su probabilidad interna de uso: p_{c_random} (el apartado 3.3 describe estos valores). C_RANDOM está pensado para explorar el espacio de entradas, pues intercambia el valor actual, c_i, por un valor al azar según una función de probabilidad uniforme. Cada componente del nuevo centro, c_i', se toma en el rango $[min_i, max_i]$ que son los valores mínimo y máximo de la $i-esima$ dimensión del espacio de entradas. Tanto min_i como max_i se pueden calcular a partir de los patrones disponibles.

Por su parte, R_RANDOM modifica los radios usando la amplitud del espacio de entradas, y aplicando de una función probabilidad uniforme. Al igual que con C_RANDOM, el número de las neuronas realmente afectadas por R_RANDOM depende de su probabilidad de uso interno p_{r_random}. El apartado 3.3 describe los valores que se deben utilizar para que el algoritmo funcione correctamente.

3.1.3. Modificación del tamaño de la capa oculta: ADDER y DELETER

EvRBF intenta determinar el tamaño correcto de la capa oculta. En este sentido, el operador ADDER incrementa el número de las neuronas ocultas creando algunas nuevas, mientras que el operador DELETER reduce dicho número eliminando neuronas.

ADDER agrega una neurona estableciendo su centro y radios al azar, usando para ello una función de probabilidad uniforme en el rango $[min_i, max_i]$. En última instancia, la presión selectiva recompensará las redes que contienen la neurona nueva (si da lugar a incremento de la función de evaluación) o las penalizará (si produce una acción negativa).

Por su parte, DELETER elimina neuronas de la RNFBR eligiéndolas totalmente azar. El número de las neuronas que se quitarán depende del valor de p_{del_many}, que determina la probabilidad de que una neurona sea eliminada. Nuevamente, el apartado 3.3 muestra el valor estimado para este parámetro.

3.2. Resto de componentes

La biblioteca EO permite la reutilización de componentes para construir nuevos algoritmos, como ocurre con EvRBF. Así, los mecanismos para seleccionar individuos para su reproducción, seleccionar los operadores a aplicar o

extraer de información del proceso evolutivo, entre otros, han sido tomados directamente de esta biblioteca. El algoritmo 1 muestra el esquema de EvRBF.

Dentro de EvRBF, *los conjuntos de entrenamiento, validación y test* se cargan en primer lugar. Estos ficheros son necesarios para estimar los rangos y dimensiones de los espacios de entrada y salida. Además, el conjunto de entrenamiento se utiliza para calcular los pesos sinápticos (véase la sección 1) y el de validación para evaluar cada individuo, una vez que se haya entrenado. Los conjuntos de entrenamiento y de validación son disjuntos. Finalmente, el conjunto de test se utiliza en para establecer la capacidad de generalización de cada individuo de la última generación.

```
Cargar conjuntos de patrones.
Crear población inicial.
Entrenar y evaluar cada individuo.
Instanciar operadores y monitores.
Instanciar el algoritmo evolutivo.
Mientras no se dé la condición de parada:
   Seleccionar individuos.
   Realizar copias de los mismos.
   Aplicar operadores a dichas copias.
   Entrenar y evaluar los individuos.
   Reemplazar los peores individuos
      de la generación actual, con
      las copias generadas.
Entrenar última generación.
Calcular su capacidad de generalización.
```

Algoritmo 1: Esquema general de EvRBF.

La función de evaluación mide la capacidad de generalización de cada individuo como la inversa de la raíz cuadrada del error cuadrático medio. Cuando se comparan dos individuos para seleccionar uno de ellos mediante el método de torneo, se entenderá que un individuo es mejor si tiene mejor *fitness* o si, teniendo exactemente el mismo valor, es menor en tamaño que el otro. En problemas de regresión y series temporales es muy improbable que dos individuos obtengan exactamente la misma medida de bondad al ser evaluados, por lo que la evolución es guiada de forma casi úncia por esta medida, sin que intervenga el tamaño de las redes.

El tamaño de la población es constante en EvRBF. Cada individuo es una RNFBR completa y la estructura de datos usada para representarla incluye métodos para ser creadas, copiadas y destruidas, para tener acceso y modificar sus componentes, y para obtener un valor de salida en función de un patrón de la entrada. La primera población se crea usando usando un determinado proceso (ver apartado 3.3), mientras las siguientes se crean mediante la eliminación de individuos de la generación actual y su reemplazamiento por individuos generados mediante operadores genéticos.

Las condiciones de parada incluidas en EvRBF permiten comprobar el número de generaciones, el umbral de error e incluso la convergencia de los individuos; estas condiciones pueden ser usadas en combinación o de forma independiente. Para este trabajo, EvRBF finaliza cuando un número prefijado de generaciones se ha alcanzado.

Finalmente, existen los operadores de *selección, reproducción y sustitución* que se aplican a las poblaciones. EvRBF utiliza el método de torneo con longitud fija como para seleccionar los individuos a reproducir, dando así una oportunidad pequeña de reproducirse a malos individuos. La reproducción se realiza mediante copia del individuo seleccionado, a la que posteriormente se aplican operadores genéticos. La sustitución clasifica a individuos según su *fitness*, quita los peores y agrega los nuevos individuos generados por la reproducción.

3.3. Parámetros de ejecución

EvRBF requiere de un conjunto de parámetros para trabajar correctamente. Actualmente, se llevan a cabo una serie de experimentos sistemáticos para establecer el mejor valor que se debe asignar a cada parámetro, aunque este estudio está fuera del alcance de este trabajo. Así, para el conjunto de experimentos descritos en el apartado 4, se han utilizado valores estándar frcuentemente usados en la literatura.

Los parámetros considerados son los siguientes: límite para las neuronas ocultas de la

primera población, métodos de inicialización centros y radios, porcentaje de la población que es sustituida, tamaño del torneo y probabilidad de uso (y tasa de uso interno, si procede) para cada operador. Las tablas 1 y 2 muestran los valores asignados a cada parámetro.

Parámetro	*Valor*
Tamaño de población	100
Generaciones	$\{10, 20, 50, 75, 100\}$
Límite de neuronas en primera generación	$1\,\%N_{Tr}$
Inicialización de centros	Patrones de entrenamiento
Inicialización de radios	Mínima distancia ponderada
Población reemplazada	30 %
Tamaño de torneo	2

Tabla 1: Parámetros de ejecución para EvRBF. Ver también tabla 2.

La tabla 1 indica que las redes de la primera generación pueden tener un número aleatorio de neuronas ocultas, en el rango de 1 al 1 % del tamaño del conjunto de entrenamiento. Para las siguientes generaciones no se impone ningún límite al número de neuronas.

Adicionalmente, los centros son inicializados usando patrones seleccionados al azar del conjunto de entrenamiento, si bien otros enfoques podrían ser considerados (como fijar centros puramente aleatorios, o el usar K-medias para seleccionar un conjunto de puntos representativo). Por otra parte, los radios son inicializados calculando la distancia del centro de la neurona que es inicializada al resto de neuronas en la red, y estableciendo su valor como 1,75 veces la distancia mínima encontrada. Otros métodos que se pueden considerar para fijar los radios incluiyen elegir un único valor aleatoriamente para cada red, elegir muchos valores aleatoriamente (uno por radio), e incluso utilizar un valor de ponderación distinto para la distancia mínima (valor que suele ir de 0,5 a 1,75).

Finalmente, en cada generación, 30 individuos son reemplazados por 30 nuevos individuos creados mediante reproducción.

La tabla 2 muestra la *tasa de aplicación* y la *probabilidad interna de uso* de cada operador. Los valores son los habituales para este tipo de trabajos, esto es, 0,4 para cada operador de recombinación, y 0,05 para los de mutación. La

Operador	*Tasa de aplicación*	*Prob. de uso interno*
X_FIX	0.4	-
X_MULTI	0.4	0.5
C_RANDOM	0.05	0.5
R_RANDOM	0.05	0.5
ADDER	0.05	-
DELETER	0.05	0.5

Tabla 2: Tasas de aplicación y probabilidad de uso interno de los operadores.

probabilidad interna de uso se utiliza par determinar qué neuronas de RBFNN se verán afectadas por la acción del operador (dichas probabilidades se corresponden con los términos p_{x_multi}, $p_{x\_average}$, p_{c_random}, p_{r_random}, and $p_{deleter}$, citados a lo largo del apartado 3.1). Para este trabajo, estos parámetros han tomado el valor 0,5, de modo que, en media, la mitad de las neuronas se ven afectas.

4. Experimentos y resultados

La serie temporal objeto de estudio mide el consumo de agua de un depósito a lo largo de 5 años y 6 meses (del 01/01/2001 al 30/06/2006). En el presente trabajo se intenta predecir el consumo durante el siguiente mes, esto es, 31 valores correspondientes a julio de 2006.

La metodología utilizada intenta emplear directamente los valores proporcionados por la serie temporal sin necesidad de aplicar técnicas habituales dentro del análisis de series temporales, tales como estacionarización de la serie mediante el cómputo de las diferencias de primer y sucesivos órdenes.

Dado que EvRBF no está diseñado para la selección de características, esto es, para determinar qué valores previos de la serie temporal influyen en los valores futuros, el primer paso ha consistido en el cálculo de la función de correlación de la serie temporal. Dicha función de correlación muestra que los tiempos más correlacionados para cada valor son los que difieren 1, 7, 14, 21, 35, 42, 364 y 734 días, esto es, el consumo de agua es bastante similar en función del día de la semana en que nos encontremos, así como de la estación del año.

Para generar y evaluar redes RBF diseñadas

Días considerados	*Patrones entrenamiento*	*Patrones test*
1,7	1819	182
1,14	1813	182
1,21	1806	182
1,35	1792	182
1,42	1785	182
35,42	1785	182
1,364	1463	182
1,7,14	1813	182
21,35,42	1785	182
1,7,14,21	1806	182
7,14,21,364,734	1902	182

Tabla 3: Particiones de tiempos consideradas (basadas en los coeficientes de correlación) y número de patrones de los ficheros de entrenamiento y test.

con EvRBF, se han creado ficheros de entrenamiento (utilizando los datos hasta 31/12/2005) y test (utilizando los datos de 01/01/2006 a 30/06/2006) usando distintas combinaciones de valores de la serie temporal. Así, la tabla 3 muestra las combinaciones de *días considerados* en el cálculo de cada valor, el número de patrones que para dicha combinación forma el conjunto de entrenamiento y el número de patrones del conjunto de test.

Para cada una de las particiones consideradas, se ha ejecutado EVRBF con diversos números de generaciones: 10, 25, 50, 75 y 100. Para cada uno de estos valores, el algoritmo se ha ejecutado 10 veces. En cada una de la ejecuciones se ha partido de poblaciones iniciales distintas, así como de distintos conjuntos de entrenamiento y de validación disjuntos entre sí. Los conjuntos de validación se han obtenido separando al azar el 25 % de los patrones del conjunto de entrenamiento.

Los resultados que se ofrecen a continuación son valores promedio a lo largo de las 10 ejecuciones realizadas para cada experimento. Dado que en cada experimento se han obtenido 100 redes en la última generación, los valores mostrados corresponden a las medias de los individuos que tenían mejor capacidad de generalización en cada experimento. La capacidad de generalización de cada individuo se ha medido como el ECM cometido en la predicción de los valores del conjunto de test, esto es, los correspondientes al período de 1/1/2006 a 30/6/2006.

Mejores resultados		
		ECM
Días considerados	Generaciones	Media $\pm$ Desv.Est.
1,7	10	$\mathbf{0.41 \pm 0.01}$
1,7	25	$\mathbf{0.41 \pm 0.01}$
1,7,14,21	75	$\mathbf{0.41 \pm 0.02}$
1,7	50	0.42 ± 0.01
1,7,14,21	25	0.42 ± 0.02
1,7,14,21	50	0.42 ± 0.02
1,7,14	50	0.42 ± 0.02
1,7	75	0.42 ± 0.01
1,7,14	25	0.43 ± 0.02
1,14	50	0.43 ± 0.01
1,7,14	100	0.43 ± 0.03
1,7,14	10	0.43 ± 0.02
1,14	75	0.43 ± 0.01
1,7,14,21	10	0.44 ± 0.02
1,14	25	0.44 ± 0.01
1,7,14,21	100	0.44 ± 0.02
1,7	100	0.44 ± 0.02
1,7,14,21	10	0.44 ± 0.02
1,14	10	0.47 ± 0.01

Muestra del resto de combinaciones		
		ECM
Días considerados	Generaciones	Media $\pm$ Desv.Est.
1,21	25	0.47 ± 0.01
1,42	25	0.51 ± 0.01
1,35	25	0.52 ± 0.01
7,14,21,364,734	50	0.70 ± 0.08
21,35,42	25	0.84 ± 0.04
1,364	10	0.97 ± 0.20
35,42	10	01.23 ± 0.03

Tabla 4: ECM obtenido en la predicción de los valores de 1/1/2006 a 30/06/2006 para las distintas combinaciones de días considerados. Ordenados de menor a mayor ECM.

La tabla 4 muestra los mejores resultados encontrados. Para aquellas combinaciones que no arrojaban buenos resultados, sólo se muestra el mejor valor obtenido, en vez de los 5 correspondientes a cada uno de los valores del número de generaciones.

Los resultados de la tabla 4 muestran que EvRBF necesita fundamentalmente los valores t_{i-1} y t_{i-7} para calcular cada valor t_i. Por su parte, el test de Student revela que no hay diferencias significativas entre los tres primeros resultados, esto es, los correspondientes al uso de los valores t_{i-1} y t_{i-7}, para 10 y 25 generaciones, y los valores t_{i-1}, t_{i-7}, t_{i-14} y t_{i-21} para 75 generaciones. De esta forma, se plantea la disyuntiva de qué red utilizar para proporcionar

Días considerados	*Generac.*	*Id. Red*	*ECM*	*Tamaño*
1,7,14,21	75	7	**0.29**	11
1,7,14,21	75	9	0.38	10
1,7	10	1	0.47	10
1,7	10	7	0.49	13
1,7	10	4	0.51	12
1,7,14,21	75	4	0.56	14
1,7,14,21	75	10	0.61	**9**
1,7,14,21	75	8	0.63	11
1,7	10	8	0.69	12
1,7	25	7	0.69	13

Tabla 5: ECM obtenido en la predicción de los valores de 1/7/2005 a 31/7/2005 para distintos días y sin tener en cuenta los valores reales conocidos para dichas fechas.

la estimación de los valores correspondientes a julio de 2006. Además, el uso de los días previos considerados implica que para el cálculo de los valores desconocidos será necesario utilizar valores estimados por la propia red. Así, para el cálculo del 1/7/2006 se podrán usar valores reales conocidos; sin embargo, para el 2/7/2006 y sucesivos tendremos que hacer uso de valores previamente estimados por la red, acumulando así el error cometido por la misma. Dado que en ninguno de los experimentos se han realizado predicciones basadas en valores estimados por la red, la elección de la mejor predicción se ha hecho observando qué red es la que mejor estima los valores de julio de 2005 pero suponiendo que sólo se conocen los valores de la serie desde 1/1/2001 a 30/6/2005.

La tabla 5 muestra el ECM cometido por las distintas redes para cada una de las configuraciones de los experimentos (recuérdese que por cada configuración surgen 10 redes, una por cada ejecución del experimento), en la tarea de predecir los valores de julio de 2005. Dado que el comportamiento de la red es totalmente determinista, sólo se realiza una ejecución con cada red y, por tanto, no se muestran desviaciones estándar. Junto al ECM de cada red, se indica el tamaño de la misma, entendido como el número de neuronas en la capa oculta.

A la vista de los resultados mostrados en la tabla 5, la decisión final ha sido utilizar la red número 7, que utiliza los tiempos t_{i-1}, t_{i-7}, t_{i-14} y t_{i-21}, y que ha surgido tras una evolución de 75 generaciones. Como se puede observar, el tamaño de dicha red no es superior al del resto de redes encontradas, por lo que el método EvRBF ha sido capaz de encontrar redes con una buena capacidad de generalización sin necesidad de sobredimensionar la capa oculta de cada red.

5. Conclusiones

En este trabajo se ha presentado el algoritmo EvRBF, diseñado para evolucionar RNFBR y aplicado a la tarea de predecir una serie temporal real.

EvRBF determina automáticamente la topología y configuración de una RNFBR. Así, EvRBF encuentra el tamaño de la red (número de neuronas ocultas) y los parámetros que configuran cada neurona: centro y radios de su función de activación. EvRBF no parte de topologías preestablecidas, no fija el número de neuronas a priori, tampoco fija un límite superior al número de neuronas, y utiliza solamente información sobre la capacidad de generalización de cada red; sólo en el caso de que dos redes tengan la misma capacidad de generalización (muy improbable en el caso de problemas de regresión y predicción de series temporales), se hace uso del tamaño de las redes para comparar las mismas, evaluarlas y hacerlas evolucionar.

EvRBF incluye operadores diseñados específicamente para tratar con RNFBR. Los operadores realizan entrecruzamiento, adición, eliminación y modificación de neuronas ocultas, así como sus respectivos componentes. Los parámetros que requiere el algoritmo para aplicar los operadores se han fijado por lo que no se precisa un proceso explícito previo a la aplicación del algoritmo a los nuevos problemas.

Las líneas futuras de investigación se centrarán en un análisis más profundo de los parámetros que hacen funcionar el algoritmo (por medio de una metodología ANOVA), el uso de la validación cruzada para establecer el *fitness* de cada nuevo individuo, así como hacer frente a este problema enfocándolo como una tarea de optimización multiobjetivo.

Agradecimientos

El presente trabajo ha sido desarrollado con la financiación de los proyectos TIN2005-08386-C05-03 y RNM-475 (Junta de Andalucía, convocatoria 2005).

Referencias

[1] C. Bishop, *Neural Networks for Pattern Recognition*, Oxford University Press, 1995.

[2] C.L. Blake and C.J. Merz., *UCI Repository of machine learning databases*, 1998.

[3] B. E. Boser, et al *A Training Algorithm for OptimalMargin Classifiers*. In Computational Learing Theory, pages 144-152, 1992.

[4] B. Burdsall and C. Giraud-Carrier. *GA-RBF: A Self- Optimising RBF Network*. Proceedings of ICANNGA'97, pages 348-351. Springer-Verlag, 1997.

[5] B. Carse and T.C. Fogarty. *Fast evolutionary learning of minimal radial basis function neural networks using a genetic algorithm*. Proceedings of Evolutionary Computing, pages 1-22. Springer-Verlag, 1996.

[6] P.A. Castillo, J. González, J.J. Merelo, V. Rivas, G. Romero and A. Prieto, *G-Prop-II: Global Optimization of Multilayer Perceptrons using GAs*, In CEC'99, pp. 2022-2027, Washington D.C., USA, 1999.

[7] P.A. Castillo et al. *G-Prop: Global optimization of multilayer perceptrons using GAs*. Neurocomputing, 35:149-163, November 2000.

[8] S. Chen et al. *Orthogonal Least Squares algorithm for learning Radial Basis Function Networks*. IEEE Trans. on Neural Networks, 2(2):302-309, 1991.

[9] E. S. Chng, S. Chen, and B.Mulgrew. *Gradient Radial Basis Function Networks for Nonlinear and Nonstationary Time Series Prediction*. IEEE Trans. on Neural Networks, 7(1):190-194, January 1996.

[10] W.S. Sarle. *Neural Network FAQ. Newsgroup: comp.ai.neural-nets.* Part 2, 1998. ftp://ftp.sas.com/pub/neural/FAQ.html.

[11] B. Fritzke. *Supervised learning with growing cell structures*. In Advances in Neural Information Processing Systems, volume 6, pages 255-262. Morgan Kaufmann, 1994.

[12] D. E. Goldberg. *Genetic algorithms in search, optimization, and machine learning*. Addison-Wesley, 1989.

[13] J. Gonzalez et al.*Expert Mutation Operators for the Evolution of Radial Basis Function Neural Networks*. LNCS 2084 , pages 538-545. IWANN'2001, June 2001.

[14] M. Grönroos,*Evolutionary Design of Neural Networks, Master of Science Thesis in Computer Science*. Dep. of Mathematical Sciences. University of Turku.

[15] C. Harpham et al. *A review of genetic algorithms applied to training radial basis function networks*, Neural Computing & Applications 13 (3) (2004) 193-201

[16] M. Kubat, *Decision trees can initialize radial-basis-function networks*, IEEE Trans. on Neural Networks 9 (1998) 813821.

[17] A. Leonardis, H. Bischof, *And efficient MDL-based construction of RBF networks*, Neural Networks 11 (1998) 963-973.

[18] D. Broomhead, D. Lowe, *Multivariable Functional Interpolation and Adaptative Networks*, Complex Systems 11 (1988) 321-355.

[19] J. Merelo, A. Prieto, *G-LVQ, a combination of genetic algorithms and LVQ, in Artificial Neural Nets and Genetic Algorithms*, D.W. Pearson, N.C. Steele and R.F. Albrecht, Edts., pp. 92-95, Springer-Verlag, ISBN 3-211-82692-0.

[20] J. E. Moody, C. Darken, *Fast learning in networks of locally tuned processing units*, Neural Computation 2 (1) (1989) 281-294.

[21] J. Platt, *A resource-allocating network for function interpolation*, Neural Computation 3 (2) (1991) 213-225.

[22] L. Prechelt, PROBEN1 – A set of benchmarks and benchmarking rules for neural network training algorithms, Tech. Rep. 21/94, Fakultät für Informatik, Universität Karlsruhe, D-76128 Karlsruhe, Germany (Sep. 1994).

[23] W. Press, S. Teukolsky,W. Vetterling, B. Flannery, *Numerical Recipes in C*, 2nd Edition, Cambridge University Press, 1992.

[24] N. Radcliffe, *Equivalence class analysis of genetic algorithms*, Complex Systems 5, no.2: 183-205.

[25] A. Rivera, J. Ortega, M. del Jesus, J. González, *Aproximación de funciones con evolución difusa mediante cooperación y competición de RBFs*, in proceedings of AEB'02, 2002, pp. 507-514.

[26] F. Ros, M. Pintore, A. Deman, J. R. Chrétien, *Automatical initialization of RBF neural networks*, Chemometrics and intelligent laboratory systems. doi: 10.1016/j.chemolab.2006.01.008.

[27] A. Roy, S. Govil, R. Miranda, *An Algorithm to Generate Radial Basis Function (RBF)-Like Nets for Classification Problems*, Neural Networks 8 (2) (1995) 179- 201.

[28] F. Schwenker, H. Kestler, G. Palm, M. Höher, *Similarities of LVQ and RBF learning*, in: Proc. IEEE International Conference SMC, 1994, pp. 646-651.

[29] F. Schwenker, H. A. Kestler, G. Palm, *Three learning phases for radial-basis function networks*, Neural Networks 14 (2001) 439-458.

[30] B. A. Whitehead, T. Choate, *Cooperative-Competitive Genetic Evolution of Radial Basis Function Centers and Widths for Time Series Prediction*, IEEE Trans. on Neural Networks 7 (4) (1996) 869-880.

[31] X. Yao, Evolving artificial neural networks, Proceedings of the IEEE, 87(9):1423-1447.

[32] R. Zhang, G.-B. Huang, N. Sundararajan, P. Saratchandran, *Improved GAP-RBF network for classification problems*, Neurocomputing. Available online. doi: doi:10.1016/j.neucom.2006.07.016.